Electronic Version
Stylesheet Version v1.1.1

Description

METHODS AND APPARATUS FOR SHARING GRAPHICAL SCREEN DATA IN A BANDWIDTH-ADAPTIVE MANNER

FIELD OF THE INVENTION

[0001] The present invention relates generally to synchronization of source node and consumer node data sets and, more particularly, to techniques and apparatus for synchronizing, in a bandwidth-adaptive manner, each of a plurality of consumer node representations of a dynamic data set with a source node representation of the dynamic data set.

BACKGROUND OF THE INVENTION

[0002] The promise of using the global computer network, colloquially referred to as the Internet, to allow many different individuals from disparate geographic and temporal regions to communicate and collaborate in real-time or near real-time remain largely unfulfilled. Differing bandwidths

of different connections result in difficulties sharing time-sensitive information. The simplest example of this effect is "screen sharing," that is, updating the screens of multiple clients such that each one mirrors a server's screen as closely as possible. Either screen updates are limited to the speed of the slowest connection, or users communicating over lower-bandwidth connections are "left behind" by those with higher-bandwidth connections. Further, in order to be useful, a system should support several simultaneous information sources and many information consumers, e.g., one or more screen broadcasts to hundreds of viewers. Additionally, a system should allow information consumers to "join late," that is, to begin receiving information from the information source at a point in time later than the beginning of the information flow.

[0003] Some attempts at solving this problem rely on a central server to analyze the information traffic and route traffic between information sources and information consumers. Although these techniques are moderately successful, use of a central server to make decisions based on the content of the routed information destroys the confidentiality of the routed information, which is unacceptable.

BRIEF SUMMARY OF THE INVENTION

[0004] The present invention enables synchronization of display data to multiple consumer nodes that adapts to available bandwidth by discarding transient states of the data set. The system fully utilizes the bandwidth available to each consumer node and is simple, efficient, and reliable. The system also has the ability to host multiple one-to-many sessions and allows consumer nodes to join an ongoing one-to-many session at any time. The system also supports end-to-end encryption of data.

[0005] In one aspect the present invention relates to a bandwidth-adaptive method for synchronizing display data between a consumer node and a source node. The source node identifies a change in local display data and creates at least one data packet representing the change in local display data. Metadata information is received from the source node which identifies a plurality of data packets that represent a state of at least a portion of the display data. At least one of the identified data packets is received from the source node. At least one of the received data packets is selected responsive to the received metadata information. The metadata information and the selected at least one data packet are transmitted to a consumer node.

[0006] In some embodiments, a request from a consumer node

for the current state of the changing display data is received before at least one of the received data packets is selected responsive to the received metadata information. In certain of these embodiments, multiple pieces of metadata information and multiple data packets are received before the consumer node request is received. In specific ones of these certain embodiments, one of the received pieces of metadata information is and at least one of the received data packets identified by the selected metadata information is transmitted to the consumer node.

[0007] In other embodiments, a plurality of the received data packets is selected responsive to the received metadata information. In certain of these other embodiments, each of the selected plurality of data packets is transmitted to the consumer node. In still other embodiments, at least one of the identified data packets is received in encrypted form, in compressed form, or in encrypted and compressed form.

[0008] In other embodiments the received metadata information or the received at least one data packet is stored in a memory device. In some of these other embodiments, at least one of the received data packets is selected responsive to the received metadata information and at least one

of the stored data packets is selected responsive to the received metadata information. In certain of these embodiments, the selected at least one of the received data packets and the selected at least one of the stored data packets is transmitted to the consumer node.

[0009] In yet further embodiments, information identifying the at least one data packet transmitted to the consumer node is stored in a memory element. In certain of these further embodiments, at least one of the received data packets is selected responsive to the received metadata information and the stored information identifying the at least one data packet transmitted to the consumer node.

[0010] In another embodiment, a bandwidth-adaptive system for synchronizing display data between a consumer node and a source node includes a source node transmitting at least one metadata packet identifying a plurality of data packets that represent the current state of changing display data and transmitting at least one of the identified data packets and a communications service in communication with the source node, the communications service selecting one of the at least one metadata packet and the at least one data packet for transmission to a first consumer node. In some embodiments, the system also includes a

first consumer node, wherein the first consumer node requests the current state of the changing data set from the communications service.

[0011] In certain of the some embodiments, the communication service selects one of the at least one metadata packet and the at least one data packet in response to the request made by the first consumer node. In others of the some embodiments, the system includes a second consumer node, wherein the second consumer node requests the current state of the changing data set from the communications service.

[0012] In further embodiments, the system includes a memory element, such as a persistent storage device. In some of these further embodiments, the communications service stores the received at least one metadata packet in the memory element. In others of these further embodiments, the communications service stores the received at least one data packet in the memory element. In still others of these further embodiments, the communications service stores in the memory element information regarding transmission of packets to a consumer node.

[0013] In yet another aspect, the present invention relates to a communications service synchronizing consumer node

representations and a source node representation of a changing display data. The communications service includes a receiving subsystem, a synchronization engine, and a transmission subsystem. The receiving subsystem receives at least one metadata packet identifying at least one data packet representing the current state of changing display data and at least one data packet identified by the received at least one metadata packet. The synchronization engine selects one of the at least one metadata packet and the at least one data packet. The transmission subsystem transmits the selected one of the at least one metadata packet and the at least one data packet. In some embodiments, the communications service also includes a memory element. In still other embodiments, the synchronization engine selects one of the at least one metadata packet and the at least one data packet in response to a request received from a consumer node.

[0014] In still another aspect, the present invention relates to a bandwidth-adaptive method for synchronizing display data between a source node and a plurality of consumer nodes. The source node identifies a first change in local display data. First metadata information identifying a first at least one data packet representing a first state of local

display data is received from the source node. The source node identifies a second change in local display data. Second metadata information identifying a second at least one data packet representing a second state of local display data is received from the source node. Third metadata information representing the difference between the first at least one identified data packet and the second at least one identified data packet is generated, the third metadata information identifying a third at least one data packet. The third metadata information and the third at least one data packet is transmitted to a consumer node.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] These and other aspects of this invention will be readily apparent from the detailed description below and the appended drawings, which are meant to illustrate and not to limit the invention, and in which:

[0016] FIG. 1 is a diagrammatic view of one embodiment of a networked system having multiple consumer nodes in communication with a source node;

[0017] FIGs. 2A and 2B are block diagrams depicting embodiments of computers useful in connection with the present invention;

[0018] FIG. 3 is a block diagram depicting packet flow in one em-

bodiment of an architecture for synchronizing data sets between a source node and a plurality of consumer nodes in a bandwidth-adaptive manner;

[0019] FIG. 4 a block diagram of an embodiment of an architecture for synchronizing data sets between a source node and a plurality of consumer nodes in a bandwidth-adaptive manner;

[0020] FIG. 5 is a diagrammatic view of a system for sharing screen data; and

[0021] FIG. 6 is a diagrammatic representation of a data structure useful in a system for sharing graphical screen data.

DETAILED DESCRIPTION OF THE INVENTION

[0022] Referring now to FIG. 1, a networked system having a source node 100 in communication with a number of consumer nodes 150, 150', 150" is depicted. As shown in FIG. 1, the consumer nodes 150, 150', 150" may communicate with the source node 100 via networks of differing bandwidth. In the embodiment shown in FIG. 1 consumer node 150 communicates with the source node 100 via a high-bandwidth network 160, such as a local area network (LAN). Consumer node 150" communicates with the source node 100 via a low-bandwidth network 180, such as a wireless network. Consumer node 150' communicates

with the source node 100 via a network 170 having bandwidth between the low-bandwidth network 180 and the high-bandwidth network 160, such as a Digital Subscriber Line (DSL) connection. Although only one source node 100 and three consumer nodes 150, 150', 150" are depicted in the embodiment shown in FIG. 1, it should be understood that the system may provide multiple ones of any or each of those components. For example, in one embodiment, the system includes multiple, logically-grouped source nodes 100, each of which may be available to provide data to a consumer node 150, 150', 150". In these embodiments, the logical group of source nodes 100 may be referred to as a "server farm" or "content farm." In other embodiments, the source node 100 is a multi-user server having a virtual frame buffer, i.e., a presentation server.

[0023] The network connections 160, 170, 180 between the consumer nodes 150, 150', 150" and the source node 100 can be local area networks (LAN), metropolitan area networks (MAN), or a wide area network (WAN) such as the Internet. The source node 100 and the consumer nodes 150, 150', 150" may connect to the networks 160, 170, 180 through a variety of connections including standard telephone lines, LAN or WAN links (e.g., T1, T3, 56 kb,

X.25), broadband connections (ISDN, Frame Relay, ATM), and wireless connections. Connections between the source node 100 and the consumer nodes 150, 159', 150" may use a variety of data-link layer communication protocols (e.g., TCP/IP, IPX, SPX, NetBIOS, NetBEUI, SMB, Ethernet, ARCNET, Fiber Distributed Data Interface (FDDI), RS232, IEEE 802.11, IEEE 802.11a, IEE 802.11b, IEEE 802.11g and direct asynchronous connections). Although shown in FIG. 1 as separate networks, networks 160, 170, 180 may be combined in a single physical network.

[0024] In many embodiments, the source node 100 and the consumer nodes 150, 150', 150" are provided as personal computer or computer servers, of the sort manufactured by the Hewlett-Packard Corporation of Palo Alto, California or the Dell Corporation of Round Rock, TX. FIGs. 2A and 2B depict block diagrams of a typical computer 200 useful as the source node 100 and the consumer nodes 150, 150', 150". As shown in FIGs. 2A and 2B, each computer 200 includes a central processing unit 202, and a main memory unit 204. Each computer 200 may also include other optional elements, such as one or more input/output devices 230a-230n (generally referred to using reference numeral 230), and a cache memory 240 in com-

munication with the central processing unit 202.

[0025] The central processing unit 202 is any logic circuitry that responds to and processes instructions fetched from the main memory unit 204. In many embodiments, the central processing unit is provided by a microprocessor unit, such as: the 8088, the 80286, the 80386, the 80486, the Pentium, Pentium Pro, the Pentium II, the Celeron, or the Xeon processor, all of which are manufactured by Intel Corporation of Mountain View, California; the 68000, the 68010, the 68020, the 68030, the 68040, the PowerPC 601, the PowerPC604, the PowerPC604e, the MPC603e, the MPC603ei, the MPC603ev, the MPC603r, the MPC603p, the MPC740, the MPC745, the MPC750, the MPC755, the MPC7400, the MPC7410, the MPC7441, the MPC7445, the MPC7447, the MPC7450, the MPC7451, the MPC7455, the MPC7457 processor, all of which are manufactured by Motorola Corporation of Schaumburg, Illinois; the Crusoe TM5800, the Crusoe TM5600, the Crusoe TM5500, the Crusoe TM5400, the Efficeon TM8600, the Efficeon TM8300, or the Efficeon TM8620 processor, manufactured by Transmeta Corporation of Santa Clara, California; the RS/6000 processor, the RS64, the RS 64 II, the P2SC, the POWER3, the RS64 III, the POWER3-II, the RS

64 IV, the POWER4, the POWER4+, the POWER5, or the POWER6 processor, all of which are manufactured by International Business Machines of White Plains, New York; or the AMD Opteron, the AMD Athalon 64 FX, the AMD Athalon, or the AMD Duron processor, manufactured by Advanced Micro Devices of Sunnyvale, California.

[0026] Main memory unit 204 may be one or more memory chips capable of storing data and allowing any storage location to be directly accessed by the microprocessor 202, such as Static random access memory (SRAM), Burst SRAM or SynchBurst SRAM (BSRAM), Dynamic random access memory (DRAM), Fast Page Mode DRAM (FPM DRAM), Enhanced DRAM (EDRAM), Extended Data Output RAM (EDO RAM), Extended Data Output DRAM (EDO DRAM), Burst Extended Data Output DRAM (BEDO DRAM), Enhanced DRAM (EDRAM), synchronous DRAM (SDRAM), JEDEC SRAM, PC100 SDRAM, Double Data Rate SDRAM (DDR SDRAM), Enhanced SDRAM (ESDRAM), SyncLink DRAM (SLDRAM), Direct Rambus DRAM (DRDRAM), or Ferroelectric RAM (FRAM).

[0027] In the embodiment shown in FIG. 2A, the processor 202 communicates with main memory 204 via a system bus 220 (described in more detail below). FIG. 2B depicts an

embodiment of a computer system 200 in which the processor communicates directly with main memory 204 via a memory port. For example, in FIG. 2B the main memory 204 may be DRDRAM.

[0028] FIGs. 2A and 2B depict embodiments in which the main processor 202 communicates directly with cache memory 240 via a secondary bus, sometimes referred to as a "backside" bus. In other embodiments, the main processor 202 communicates with cache memory 240 using the system bus 220. Cache memory 240 typically has a faster response time than main memory 204 and is typically provided by SRAM, BSRAM, or EDRAM.

[0029] In the embodiment shown in FIG. 2A, the processor 202 communicates with various I/O devices 230 via a local system bus 220. Various busses may be used to connect the central processing unit 202 to the I/O devices 230, including a VESA VL bus, an ISA bus, an EISA bus, a MicroChannel Architecture (MCA) bus, a PCI bus, a PCI-X bus, a PCI-Express bus, or a NuBus. For embodiments in which the I/O device is an video display, the processor 202 may use an Advanced Graphics Port (AGP) to communicate with the display. FIG. 2B depicts an embodiment of a computer system 200 in which the main processor 202

communicates directly with I/O device 230b via HyperTransport, Rapid I/O, or InfiniBand. FIG. 2B also depicts an embodiment in which local busses and direct communication are mixed: the processor 202 communicates with I/O device 230a using a local interconnect bus while communicating with I/O device 230b directly.

[0030] A wide variety of I/O devices 230 may be present in the computer system 200. Input devices include keyboards, mice, trackpads, trackballs, microphones, and drawing tablets. Output devices include video displays, speakers, inkjet printers, laser printers, and dye-sublimation printers. An I/O device may also provide mass storage for the computer system 200 such as a hard disk drive, a floppy disk drive for receiving floppy disks such as 3.5-inch, 5.25-inch disks or ZIP disks, a CD-ROM drive, a CD-R/RW drive, a DVD-ROM drive, tape drives of various formats, and USB storage devices such as the USB Flash Drive line of devices manufactured by Twintech Industry, Inc. of Los Alamitos, California.

[0031] In further embodiments, an I/O device 230 may be a bridge between the system bus 220 and an external communication bus, such as a USB bus, an Apple Desktop Bus, an RS-232 serial connection, a SCSI bus, a FireWire bus, a

FireWire 800 bus, an Ethernet bus, an AppleTalk bus, a Gigabit Ethernet bus, an Asynchronous Transfer Mode bus, a HIPPI bus, a Super HIPPI bus, a SerialPlus bus, a SCI/LAMP bus, a FibreChannel bus, or a Serial Attached small computer system interface bus.

[0032] General-purpose desktop computers of the sort depicted in FIGs. 2A and 2B typically operate under the control of operating systems, which control scheduling of tasks and access to system resources. Typical operating systems include: MICROSOFT WINDOWS, manufactured by Microsoft Corp. of Redmond, Washington; MacOS, manufactured by Apple Computer of Cupertino, California; OS/2, manufactured by International Business Machines of Armonk, New York; and Linux, a freely-available operating system distributed by Caldera Corp. of Salt Lake City, Utah, among others.

[0033] In some embodiments the consumer node 150, 150', 150" is a mobile device, such as a JAVA-enabled cellular telephone or personal digital assistant (PDA), such as the i50sx, i55sr, i58sr, i85s, i88s, i90c, i95cl, or the im11000, all of which are manufactured by Motorola Corp. of Schaumburg, Illinois, the 6035 or the 7135, manufactured by Kyocera of Kyoto, Japan, or the i300 or i330, manufac-

tured by Samsung Electronics Co., Ltd., of Seoul, Korea. In other embodiments in which the client device 140 is mobile, it may be a personal digital assistant (PDA), such as the Tungsten W, the VII, the VIIx, the i705, or a combination PDA/telephone device such as the Treo 180, Treo 270 or Treo 600, all of which are manufactured by palmOne, Inc. of Milpitas, California.

[0034] In these embodiments, the consumer nodes 150, 150', 150" connect to the source node 100 using any one of a number of well-known protocols from the GSM or CDMA families, such as W-CDMA. These protocols support commercial wireless communication services and W-CDMA, in particular is the underlying protocol supporting i-Mode and mMode services, offered by NTT DoCoMo.

[0035] FIG. 3 depicts a block diagram of a system for synchronizing a data set between the source node 100 and a plurality of consumer nodes 150, 150', 150", as well as the packet flow in a system during operation. As shown in FIG. 3, the system includes a communications service 300 with which the source node 100 and the plurality of consumer nodes 150, 150', 150" communicate. The source node 100 and the consumer nodes 150, 150', 150" may be located behind respective firewalls (not shown in FIG.

3). The source node 100 and the consumer nodes 150, 150', 150" make outgoing socket connections to the communications service 300. In some embodiments, the communications service 300 maintains state for each connection to a consumer node 150, 150', 150" in the form of socket descriptors. In other embodiments, the source node 100 and the communications service 300 may be provided as the same physical device. In these embodiments, the source node 100 and the communications service 300 operate on the same hardware in a time-shared manner. Data may be shared between the source node 100 and the communications service 300 using any one of a number of techniques, such as pipe objects or shared memory.

[0036] The source node 100 codes the current state of a dynamic data set, such as screen data, as a set of data packets. In some embodiments, this coding process is straightforward. For example, in the case where the dynamic data set is screen data, data packets may be coded by storing pixel values for a predetermined portion of the screen in the data packet. In some embodiments, the source node 100 compresses the data stored in the data packets. In still other embodiments, the source node 100 encrypts the

data stored in the data packets. In still further embodiments, the source node 100 both encrypts and compresses data stored in the data packets. As the dynamic data set changes, the source node updates the set of data packets comprising the current state of the data set.

[0037] The source node 100 transmits the current state of the dynamic data set to the communications service 300 in a bandwidth-adaptive manner. In one embodiment, this is achieved by requiring the source node 100 to possess a transmission token before beginning transmission of the current state of the data set. In this embodiment, the source node 100 and the communications service exchange a limited number of transmission tokens, e.g., five. In other embodiments, the communication service 300 transmits a message to the source node 100 to notify the source node 100 when it can send another data set update.

[0038] As shown in FIG. 3, the communications service 300 may also include a data storage element 310, such as random-access memory, a disk drive, a disk array, a rewriteable optical drive, or some other form of memory element that allows access to stored data. The storage element 310 enables the communications service 310 to store metadata

information and data packets received from the source node 100 in between update requests from various consumer nodes 150, 150', 150". In addition, the storage element 310 can be used to maintain a historical record of metadata information and data packets transmitted from the source node 100. In other embodiments, the storage element 310 may also store the data packets transmitted to a respective consumer node 150, 150', 150".

[0039] The source node 100 creates metadata information that identifies each of the data packets representing the current state of the dynamic data set. In the embodiment shown in FIG. 3, the metadata information comprises a metadata packet 310, 320, 330. Metadata packet 310 is created at time t1, and indicates that the state of the dynamic data set at time t1 is represented by data packet 0, data packet 1, and data packet 2. Similarly, metadata packet 330 indicates that state of the dynamic data set at time t2 is represented by data packet 0, data packet 4, and data packet 5. In other embodiments, instead of creating metadata packets that store metadata information, metadata information is included in data packets. For example, each data packet comprising a data set update may include a "metadata information header" identifying

the update set with which the data packet is associated.

[0040] As shown in FIG. 3, the source node 100 transmits metadata information 310 to the communications service 300 followed by the data packets identified by the metadata information 310. Thus, the source node 100 transmits to the communications service 300 data packet 0 312, data packet 1 314, and data packet 2 316 following the metadata packet 310. At time t2, the source node 100 transmits to the communications service 300 metadata packet 320, which indicates that the state of the data set at time t2 is represented by data packet 0, data packet 3, and data packet 4. The source node 100 then transmits data packet 3 322 and data packet 4 334 to the communications service 300. The source node 100 does not retransmit data packet 0 to the communications service 300 since that data packet was transmitted in connection with the first metadata packet 310. Similarly, at time t3 the source node 100 transmits to the communications service 300 a metadata packet 330 that indicates the current state of the dynamic data set is represented by data packet 0, data packet 4, and data packet 5. Since the source node 100 already transmitted data packet 0 to communications service 300 following the first metadata

packet 310 and data packet 4 following the second metadata packet 320, the source node 100 only transmits data packet 5 332 following the third metadata packet 330.

[0041] As described above in connection with flow control between the source node 100 and the communications service 300, flow control between the consumer nodes 150, 150', 150" and the communications service 300 may be token-based or message-based. For ease of reference, the remaining description will assume that the flow control method is based on messages. However, the same advantages of the invention can be obtained in a system relying on transmission tokens.

[0042] FIG. 3 depicts an embodiment of a system in which consumer node 150, communicates with the communications service 300 via a high-bandwidth connection. In this case, the consumer node 150 requests data set updates frequently enough that the communication service 300 transmits to the consumer node 150 a stream of metadata information and data packets identical to the stream of metadata information and packets received by the communications service 300 from the source node 100. Also as shown in FIG. 3, the consumer node 150", which communicates with the communications service 300 via a low-

bandwidth connection, requests data set updates less frequently and, therefore, receives a different stream of packets from the communications service 300 than the communications service 300 receives from the source node 100. As shown in FIG. 3, the communications service 300 transmits the first metadata packet 310 and data packets 0-3, 312, 314, 316 to the consumer node 150". The next metadata packet received by the consumer node 150" is the third metadata packet 330, which indicates that the state of the dynamic data set is represented by data packet 0, data packet 4, and data packet 5. Since the consumer node 150" has not yet received data packet 4 and data packet 5, the communications service 300 transmits those data packets to the consumer node 150".

[0043] FIG. 3 also depicts the packet stream sent to a consumer node that "joins late." As shown in FIG. 3, a consumer that joins at time t3 will receive the third metadata packet 330, as well as all the data packets identified by the third metadata packet. The data packets transmitted to the consumer node 150, 150', 150" by the communications service 300 may be retrieved from the storage element 310, recently received from the source node 100, or some combination of the two.

[0044] Delivery of data set updates from the communications service 300 may be performed using a "push" model, a "pull" model, or an "atomic push" model. In the "push" models, the communication service 300 transmits metadata information and data packets to the consumer node 150, 150', 150". The difference between the "push" model and the "atomic push" model is that, in the "atomic push" model, the communications service 300 commits to transmit every data packet identified by transmitted metadata information before beginning transmission of another data set. There is no such commitment in the "push" model, which means that data packets not successfully transmitted from a previous data set update may never be sent to the consumer node 150, 150', 150". In the "pull" model, the consumer node 150, 150', 150" receives from the communications service 300 the metadata information and then requests specific data packets from the communications service 300.

[0045] In certain embodiments, the information in metadata packets is encoded incrementally. In these certain embodiments, the "wire" representations of metadata packets may differ despite the fact that they encode the same information. A short example shows why this is the case.

Over time, the source node 100 sends three metadata packets to the communications service 300. The contents of the metadata packets are sets of data packet numbers (1, 2, 3), (2, 3, 4) and (3, 4, 5). On the "wire," each set is represented as a delta from the previous set. Thus, the source node 100 transmits the following metadata packets to the communications service 300: (1, 2, 3), (-1, +4) and (-2, +5), where '-' and '+' indicate removal or addition of a packet number from/to the previous set. If a consumer node 150 skips the contents of the second metadata packet, it receives metadata information describing sets (1, 2, 3) and (3, 4, 5). On the "wire," these two sets are represented incrementally as (1, 2, 3) and (-1, +4, -2, +5). While the source node 100 transmitted the contents of the second metadata packet to the communications service 300 as (-2, +5), the communications service 300 transmitted the same information to the consumer node 150 as (-1, +4, -2, +5).

[0046] FIG. 4 depicts another embodiment of a system for synchronizing a data set between a source node 100 and one or more consumer nodes 150, 150', 150" that includes multiple communications services 300, 300', 300" (generally referred to as 300). As shown in FIG. 4, the

source node communicates with more than one communications service 300. Similarly, each consumer node 150, 150', 150" may also communicate with one or more communication services 300. The communication services 300 also communicate in a peer-to-peer fashion among themselves.

[0047] In this embodiment, each pair of communication services 300 agrees between themselves on a direction for data flow. For example, communication service 300 and communication service 300' may agree between themselves that, for the purposes of their point-to-point link, communication service 300 is the "sender" and communication service 300' is the "receiver," meaning that the "sender" will perform the role of the communication service 300 described in connection with FIG. 3 and the "receiver" will perform the role of the consumer node 150 described in connection with FIG. 3. The communication server 300', however, will perform the role of a "sender" when communicating with consumer nodes 150, 150', 150"

[0048] The present invention may be provided as one or more computer-readable programs embodied on or in one or more articles of manufacture. The article of manufacture

may be a floppy disk, a hard disk, a compact disc, a digital versatile disc, a flash memory card, a PROM, a RAM, a ROM, or a magnetic tape. In general, the computer-readable programs may be implemented in any programming language. Some examples of languages that can be used include C, C++, C#, or JAVA. The software programs may be stored on or in one or more articles of manufacture as object code.

[0049] EXAMPLES

[0050] The following examples of content-sharing systems are intended to illustrate the various ways in which the described systems and methods can be used and not to limit the scope of the described invention.

[0051] EXAMPLE 1

[0052] The described systems and methods can be used to implement a system for sharing screen data that allows several client machines to display the screen data from a single server. This system is useful in a number of broadcast or "multicast" contexts and, in particular, it is useful in a conferencing context to allow multiple individuals to view the same graphical data during the conference.

[0053] FIG. 5 depicts diagrammatically a system for sharing

screen data. As shown in FIG. 5, a host server 100 monitors its screen state. In the embodiment shown in FIG. 5, the host server 100 subdivides its screen into 12 tiles, although any number of tiles may be used to fully represent the screen of the host server 100. In some embodiment the tiles are each the same size, that is, each tile represents the same number of screen pixels. In other embodiments, such as the embodiment shown in FIG. 5, some of the tiles have sizes different from other tiles. In still other embodiments, a tile may overlap another tile or, as shown in FIG. 5, tiles may by non-overlapping.

[0054] As shown in FIG. 5, the host server's previous screen 500 is represented by a first set of tiles (not shown), which are coded as a first set of data packets: 13, 14, 3, 4, 15, 6, 7, 8, 17, 10, 11, and 12. If the host server 100 possesses a transmission token, it transmits these twelve data packets to the communications server 200, as described above.

[0055] At a second point in time, the host server's screen 510 has changed. The host server 100 identifies the particular tiles that have changed states, and creates a coded packet for each tile that has changed, i.e., data packets 19, 20, 21, and 22. If the host server 100 did not possess a transmission token but now receives one, the host server 100

will transmit the updated twelve data packets to the communications server 200, i.e., data packets 13, 14, 3, 4, 15, 19, 20, 17, 21, 22, and 12. If the host server has already transmitted the data packets representing the state of the screen 500, then the host server 100 need only transmit to the communications server 200 data packets 19, 20, 21, and 22. In this manner, transmission of screen data between the host server 100 and the communications server 200 is performed in a bandwidth-adaptive manner.

[0056] In some embodiments, the host server 100 encrypts the data packets transmitted to the communications server 200. In other embodiments, the host server 100 compresses the data packets sent to the communications server 200. In still other embodiments, the host server 100 both encrypts and compresses the data packets.

[0057] In many embodiments, the communications server 200 maintains a copy of each tile that comprises the most recent state of the server node screen. In some embodiments, each tile is associated with a timestamp when transmitted to the communication service 200. In other embodiments, each tile is associated with a number that monotonically increases with each new tile transmitted to the communications service 300.

[0058] The communications server 200 composes an update for a viewer node 150 as often as the bandwidth of the network connecting the viewer node 150 to the communications server 200 allows. As shown in FIG. 5, the viewer's screen 520 displays screen data from a point in time before the host's previous screen 500. That is, the host server's display data has changed twice (represented by screen 500 and screen 510) since the last time the viewer node 150 has requested an update. Data packet array 570 shows the data packets comprising the screen data currently displayed by the viewer node 150. Data packet array 590 depicts the data packets that the communications server 200 must transmit to the viewer node 150 in order to update the viewer's screen 520 to the state of the host's screen 510. As described above, the communications server 200 transmits metadata information to the viewer node 150 identifying eight data packets: data packets 13, 14, 15, 19, 20, 17, 21, and 22. In some embodiments, the metadata information explicitly identifies which tile replaces which other tile, perhaps by describing the position of the new tile. The communications server 200 then transmits the packets representing the new tiles to the viewer node.

[0059] In another embodiment, the communication service 200 responds to an update request from the viewer node 150 by transmitting to the viewer node 150 every data packet having a timestamp newer than the timestamp of the viewer's screen. In some of these embodiments, the communication service 200 does not fully receive and store a set of data packets comprising a screen update before sending the update to the viewer node 150. In these embodiments, the communications service 300 sets the timestamp for each packet identified by metadata information as comprising the screen update to the same value. Then, as data packets arrive the communications service 300 streams those packets to the viewer node 150.

[0060] In one particular embodiment, metadata information is formatted into packets and metadata packets are associated with monotonically increasing numbers. As described above, each metadata packet describes the set of tiles comprising the current screen display state. In this embodiment, the communications service 300 stores, for each consumer node 150, the number of the latest metadata packet that has been transmitted to that consumer node 150, as well as the set of all data packets that have

been delivered to the consumer node. When the communications service 300 determines that it is time to send an update to a consumer node 150, or upon receiving a request from a consumer node 150 for a screen update, the communications service first determines if the latest metadata packet (that is, the metadata packet having the highest number associated with it) has been transmitted to the consumer node 150. If not, the communications service 300 transmits the most recent metadata packet to the consumer node 150. The communications service 300 also transmits the set of data packets identified by the metadata packet, unless a particular data packet has already been transmitted to the consumer node 150.

[0061] In another embodiment, the set of tiles (i.e., data packets) that must be transmitted to a consumer node is computed by associating each tile with a timestamp and identifying all visible tiles whose timestamps are newer than the newest tile already received by the consumer node 150. FIG. 6 depicts diagrammatically a data structure enabling efficient replacement of a display tile by the communications service 300 and given a timestamp, identification of which tiles are out-of-date with respect to a given consumer node 150. All tiles comprising a screen 610 are

stored in a doubly-linked list 620 sorted by timestamp and indexed by tile location in the screen. As shown in FIG. 6, when new tile 29 overwrites old tile 19, tile 19 is removed from the list and new tile 29 is inserted at the head of the list. When a viewer node 150 requests a screen update, the communications service 300 iterates through the list of tiles 620 and transmits tiles to the viewer node until it encounters a tile with a timestamp older than the newest tile on the viewer node screen.

[0062] EXAMPLE 2

[0063] In another example the described synchronization systems and methods are used to implement a chat system. In this system, a chat participant adds text or other content to an on going session and identifies the added content as a data packet. In one embodiment, the participant also associates a timestamp with the added content. The participant then transmits metadata information identifying the current state of the chat. In one embodiment, the metadata information identifies the current state of the chat session as the recently added packet together with every previous data packet added to the chat.

[0064] The participant transmits the metadata information together with a data packet representing the recently added

content. In one embodiment, the metadata information and data packet are pushed to a receiving node, as described above. Recipients of the metadata information and data packet merge the received data packet with chat data packets already received in the order the data packets are received. In another embodiment, the recipient merges the chat data packets based on the time the data packet was sent. In still another embodiment, the recipient merges the data packets based on the timestamp associated with the data packet.

[0065] A "late joiner" to the chat session will receive metadata information identifying all data packets representing the chat session. The late joiner will either request (i.e., pull) or be sent (i.e., push) all the data packets identified by the metadata information and will display in them in timestamp order.

[0066] EXAMPLE 3

[0067] In another example, the synchronization systems and methods described above may be used to implement a remote presentation system. In this example, a presenter converts a slide presentation into a series of page-by-page images. As the presenter displays a slide, the page image representing that slide is transmitted to all

viewers. In many embodiments, each slide is represented by multiple data packets.

[0068] In this example, the presenter atomically pushes the currently displayed slide by atomically pushing metadata information identifying each data packet representing the slide and pushing each data packet not yet transmitted to the receiver. The presenter may also push the previous slide and the next slide. In further embodiments, viewers may "pull" other pages in the presentation if extra bandwidth is available. Information that may be inserted into a laser pointer data packet includes, x coordinate, y coordinate, time, document, or page number.

[0069] In addition to multicasting slide presentation, this exemplary embodiment may be used to share other page-based documents. This exemplary embodiment may also support a "laser pointer" feature in which the position of a presenters "laser pointer" is also transmitted to all viewers, allowing the presenter to direct viewer's attention to areas of interest in the document.

[0070] EXAMPLE 4

[0071] In still another example, the synchronization methods and systems described above may be used to implement a system allowing multiple users to annotate a document. In

this example, each annotation is represented by a data packet. Annotation data packets may include information regarding the time the annotation was made and by whom. Other annotation data packet information may include the document on which the annotation is made, the page number of the document on which the annotation is made, the pen used to make the annotation, the x coordinate of the annotation, or the y coordinate of the annotation.

[0072] In this example, the metadata information identifies all annotation data packets. In this manner, a "late joiner" will receive all annotations made to the document.

[0073] While the invention has been shown and described with reference to specific preferred embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention as defined by the following claims.